A 2D Immersed Interface Vortex Particle-Mesh method

T. Gillis[a,*], Y. Marichal[a], G. Winckelmans[a], P. Chatelain[a]

[a]*Institute of Mechanics, Materials and Civil Engineering, Université catholique de Louvain*
1348 Louvain-la-Neuve, Belgium

Abstract

We present an Immersed Interface Vortex Particle-Mesh (IIVPM) method in 2D and validate our approach. Unlike other VPM-based approaches, it converges in time and space up to the boundary, and hence it offers a sharp and accurate way to take the presence of obstacles into account. We first present the methodology and investigate its convergence. We then apply it to the benchmark of the impulsively started flow around a cylinder at various Reynolds numbers (from Re = 550 to Re = 40 000) and, in particular, we highlight the gain in accuracy brought by the sharp interface treatment.

Keywords: Immersed Interface Method, Vortex Particle-Mesh method, Bluff-body flows, Sharp interface treatment, Dirichlet boundary condition for diffusion

1. Introduction

Vortex methods have been gaining popularity for many advection-dominated flow simulations where the vorticity is compact, such as wake studies at high Reynolds numbers and external aerodynamics. However, taking the presence of a body into account remains a challenge. The current state of the art in the treatment of inner bodies for flow problems, on regular grids, is split between the *discrete* and *continuous forcing* approaches, a differentiation introduced by Mittal and Iaccarino [1]. The *continuous forcing* approach relies on the addition of a mathematical artefact inside the Navier-Stokes equations to impose the boundary condition. The first and most widely used method in this category is the Immersed Boundary (IB) approach, initially proposed by the pioneering work of Peskin [2] and renowned for its proven robustness [3, 4]. The IB method discretizes additional forcing terms due to the immersed body on several grid points using regularized delta functions, hence mollifying the inner interface. Some recent developments propose a harmonic extension in the body, and therefore an additional Poisson equation to solve, which, coupled with the use of symmetric delta function, increases the order of convergence of the method [5]. Another widely used approach in this class of methods is the Brinkman penalization, which uses a Lagrangian relaxation of the body constraint $\mathbf{u} = \mathbf{u}_b$ [6, 7] to impose the inner boundary condition.

The second class of methods, the *discrete forcing* approach, modifies the discretization of the Navier-Stokes equations in order to take an immersed body into account. In this category, let us first mention the Ghost-Cell Method, that uses ghost cells to extend the fields inside the body by means of a polynomial extrapolation, see [1] and references therein. We also refer to the Ghost-Fluid approach, which uses the combination of a Neumann and Dirichlet boundary condition to build an extension inside the body, see [8] and references therein. Finally, still in the *discrete forcing* category, there is the Immersed Interface Method (IIM) [9–11], which uses modified Taylor series to take the inner body into account. The computation of the jump

*Corresponding author
Email address: thomas.gillis@uclouvain.be (Thomas Gillis)

values across the interface is then based on both the bulk fluid values and the boundary conditions. It can therefore be seen as a hybrid approach between the Ghost-Fluid and the Ghost-cell methods. The IIM offers a proven high order and sharp treatment of boundaries and does not require the introduction of any mollification of the body [12, 13].

The transposition of these approaches to the Vortex Particle-Mesh (VPM) framework is not straightforward and has been partially studied. In the *continuous forcing* approach, recent works directly applied the Brinkman penalization framework to VPM method [14, 15], while others propose to iterate on the body constraint in order to converge the vorticity-velocity fields enforcing the no-slip condition at the body surface, and therefore improve the solution quality [16, 17]. However, this approach is only first order at the boundary and implies multiple resolutions of the Poisson equation on the computational domain and inside the inner body to perform the penalization iterations. In the other family of method, the *discrete forcing* approach, the Boundary Element Method (BEM), also called panel method, has been coupled with the Vortex Particle framework, see e.g. [18, 19]. The convection and diffusion of the vorticity field is first handled discarding the generation of vorticity at the wall. Then, following Lighthill [20], the vortex sheet associated to the residual slip velocity is diffused into the flow (viscous splitting approach). Let us also mention the work of Cottet and Poncet [21] that proposes a similar diffusion splitting method, applied to body-fitted grids and immersed boundary approach.

To the best of the authors' knowledge, none of the proposed approaches, transposed to the VPM framework, offers a sharp and accurate treatment of the immersed boundary, while also preserving the temporal convergence order. Indeed, the above approaches either smear the immersed object interface, as in the iterative Brinkman penalization technique, which degrades the spatial accuracy at the wall or they split the diffusion operator, which limits the temporal convergence to first order, as in the VPM-BEM approach.

Nevertheless, some recent works have extended the Immersed Interface grid-based approach to the VPM approach. Marichal et al. [22] have developed an IIM-based unbounded Poisson solver around complex geometries and they have extended the particle-mesh interpolation to the immersed interface tool in [23]. Gillis et al. [24] have also presented an efficient unbounded FFT-based Poisson solver for inner boundaries using IIM, that decomposes the problem using the Sherman-Morrison-Woodbury formula [25–27].

These works have paved the way for an Immersed Interface Vortex Particle-Mesh (IIVPM) solver for incompressible flows around inner bodies. In this work, we aim at a sharp and second order IIVPM solver which additionally conserves the temporal convergence order. This paper is structured as follows: in Section 2, we first briefly recall the Vortex Particle-Mesh method as a whole, including the specific aspects of the particle-mesh interpolation and of the Biot-Savart law solver. Then, we introduce the Immersed Interface techniques and we expose their use in the present context. Finally, we validate the resulting IIVPM method in Section 3 and assess its performances on the benchmark case of an impulsively started cylinder in Section 4. A conclusion of the presented work and future perspectives are presented in Section 5.

2. Methodology

This work presents a combination of the Vortex Particle-Mesh (VPM) method with Immersed Interface (II) tools in order to take an immersed body into account. Hereunder, we first briefly describe the VPM approach (Section 2.1) and then we introduce the Immersed Interface method in Section 2.2. Finally, we present the IIVPM methodology as a whole in Section 2.3 and then, step by step: the Poisson solver in Section 2.4, the handling of the diffusion in Section 2.5 and the particle-mesh interpolation in Section 2.6 and Section 2.7.

2.1. The 2D Vortex Particle-Mesh method

We briefly present the Vortex Particle-Mesh method as used in this work, based on the velocity ($\mathbf{u}$)-vorticity ($\boldsymbol{\omega} = \omega\, \mathbf{e}_z = \nabla \wedge \mathbf{u}$) formulation of the 2D Navier-Stokes equations for an incompressible flow ($\nabla \cdot \mathbf{u} = 0$):

$$\frac{D\omega}{Dt} = \nu \nabla^2 \omega \ , \tag{1}$$

where $\frac{D}{Dt} = \frac{\partial}{\partial t} + \mathbf{u} \cdot \nabla$ denotes the Lagrangian derivative and ν is the kinematic viscosity, see [28, 29] and references therein. We here have $\mathbf{u} = \mathbf{U}_\infty + \mathbf{u}_\omega$, where $\mathbf{u}_\omega$ is recovered from the vorticity field through a Poisson equation

$$\nabla^2 \psi_\omega = -\omega \qquad\qquad \mathbf{u}_\omega = \nabla \wedge \left(\psi \mathbf{e}_z\right) \ . \tag{2}$$

A Vortex Particle-Mesh (VPM) method discretizes the vorticity field by means of particles, characterized by a position $\mathbf{x}_p$, a material surface S_p and a strength $\boldsymbol{\alpha}_p = \int_{S_p} \boldsymbol{\omega} d\mathbf{x}$. An integration of Eq. (1) over a particle surface, through a mid-point quadrature with the assumption of the particle discretization, produces the following system

$$\frac{d}{dt}\begin{bmatrix} \mathbf{x}_p \\ \boldsymbol{\alpha}_p \end{bmatrix} = \begin{bmatrix} \mathbf{u}_p \\ \nu \left(\nabla^2 \boldsymbol{\omega}\right)_p \, S_p \end{bmatrix} \ . \tag{3}$$

In the framework of VPM methods, the right hand side of Eq. (3) is evaluated on the mesh, therefore improving the computational efficiency. The particle-mesh interpolation typically relies on high order interpolation formulas [30, 31]. Once the particle quantities have been interpolated onto the mesh, one can use standard finite differences to evaluate the differential operators on the right hand side of Eq. (3). The Poisson problem of Eq. (2) can also be solved using the mesh, by means of highly efficient FFT-based solvers [32, 33]. Afterwards, the right hand side is interpolated back on the particles to integrate Eq. (3). The time integration scheme, on the particles, is a standard Runge-Kutta scheme; the low-storage RK3 scheme proposed by Williamson [34] is used in this work.

The time step is first stability-constrained by the diffusion; this is embodied by the Fourier number, i.e. $r = \frac{\nu\, \Delta t}{h^2} < r_{max}$, where r_{max} is given by the chosen time and space discretization scheme. The particle treatment of the convection imposes an accuracy condition instead, based on the Lagrangian deformation, usually measured using two criterions

$$\Delta t \left\| \boldsymbol{S} \right\|_1 \leq C_{\mathrm{LCFL,S}} \qquad \text{and} \qquad \Delta t \left\| \boldsymbol{\omega} \right\|_1 \leq C_{\mathrm{LCFL},\omega} \ , \tag{4}$$

where $\boldsymbol{S} = \frac{1}{2}\left[\nabla \mathbf{u} + (\nabla \mathbf{u})^T\right]$ is the strain tensor and $C_{\mathrm{LCFL,S}} \simeq C_{\mathrm{LCFL},\omega} \simeq 0.15 \ ... \ 0.3$ are typical values. To prevent particle clustering and depletion, due to the Lagrangian deformation, the VPM method relies on a procedure called remeshing [28, 29], which consists in the periodic redistribution of the particle set onto the mesh. Finally, we refer to [32] for an extensive description of the VPM method and its implementation for large parallel architectures.

2.2. Immersed interface framework and wall-data computation

We here propose a brief presentation of the Immersed Interface Method (IIM), as applied in this work. For a more detailed information, we refer to [9–11, 22–24] and the references therein. We start with a presentation of the basis in 1D and then detail the application in 2D.

2.2.1. General considerations

The IIM relies on the handling of discontinuities in finite difference stencils. The discontinuities are incorporated into the stencil using a jump-corrected Taylor series. As an example, let us consider a one-dimensional function $u(x) = u^{(0)}(x) \in C^{\infty}\left(\mathbb{R} \setminus \{x_\alpha\}\right)$ and describe the IIM for the evaluation of the second derivative (required for the Laplacian evaluation in 2D, as will be seen below), as shown on Fig. 1. The function $u(x)$ and its derivatives, $u^{(\cdot)}(x)$, may be discontinuous at the interface,

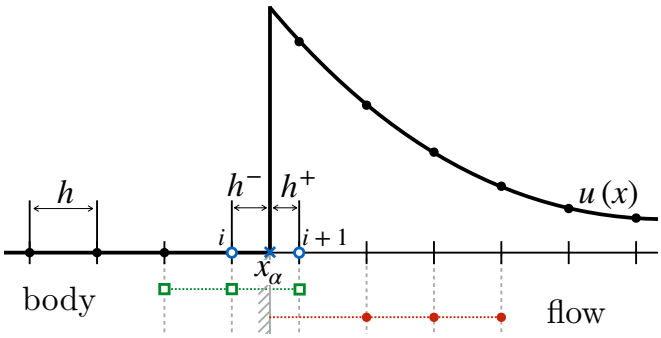


Figure 1: **1D Immersed interface scheme** The standard 3-point stencil (□) intersects the interface and generates a control point (×), discontinuity affected points (○) and primary interpolation points (●)

$x_\alpha \in \left]x_i\ ;\ x_{i+1}\right[$. Using a generalized Taylor series, we write a modified finite difference scheme at x_i and x_{i+1} for the second derivative, $u^{(2)}$, still valid across the interface (x_α)

$$u^{(2)}\left(x_i\right) = \frac{1}{h^2}\left[u\left(x_{i-1}\right) - 2u\left(x_i\right) + \left(u\left(x_{i+1}\right) - J_\alpha^+\right)\right] + \mathcal{O}\left(h^2\right) \tag{5}$$

$$u^{(2)}\left(x_{i+1}\right) = \frac{1}{h^2}\left[\left(u\left(x_i\right) + J_\alpha^-\right) - 2u\left(x_{i+1}\right) + u\left(x_{i+2}\right)\right] + \mathcal{O}\left(h^2\right) \tag{6}$$

where we define the jumps as

$$J_\alpha^+ = \left[u^{(0)}\right]_\alpha + \frac{h^+}{1!}\left[u^{(1)}\right]_\alpha + \frac{\left(h^+\right)^2}{2!}\left[u^{(2)}\right]_\alpha + \frac{\left(h^+\right)^3}{3!}\left[u^{(3)}\right]_\alpha + \mathcal{O}\left(h^4\right) \tag{7}$$

$$J_\alpha^- = \left[u^{(0)}\right]_\alpha + \frac{h^-}{1!}\left[u^{(1)}\right]_\alpha + \frac{(h^-)^2}{2!}\left[u^{(2)}\right]_\alpha + \frac{(h^-)^3}{3!}\left[u^{(3)}\right]_\alpha + \mathcal{O}\left(h^4\right) \tag{8}$$

with $h^+ = \left(x_{i+1} - x_\alpha\right)$ positive, $h^- = \left(x_i - x_\alpha\right)$ negative and the jumps in the kth derivative:

$$\left[u^{(k)}\right]_\alpha = u^{(k)}\left(x_{\alpha^+}\right) - u^{(k)}\left(x_{\alpha^-}\right) \quad . \tag{9}$$

Without loss of generality, we assume that $x < x_\alpha$ represents Ω_b. Inside Ω_b, all derivatives are supposed to be known since we know the desired solution inside the obstacle; for instance a null field as in Fig. 1. Outside Ω_b, the evaluation of the value and derivatives at the wall, required by Eq. (9), uses one-sided interpolation schemes and eventually the boundary condition (in this example, a Dirichlet boundary condition: $u_\alpha = u\left(x_\alpha^+\right)$),

$$u^{(k)}\left(x_{\alpha^+}\right) = S_\alpha^k u_\alpha + S_1^k u\left(x_{i+2}\right) + S_2^k u\left(x_{i+3}\right) + S_3^k u\left(x_{i+4}\right) + \mathcal{O}\left(h^{4-k}\right) \quad , \tag{10}$$

which is in agreement with the global accuracy needed by the corrected scheme and the second derivative evaluation. The intersection between the mesh and the body surface is called “control point", the points x_i and x_{i+1} are called “discontinuity affected points" and the mesh-points needed by Eq. (10) are called “primary interpolation points".

Other variants of the presented methodology are also used in this work when we evaluate first degree operators, $\nabla\cdot$ or $\nabla\wedge$ for example, or when we do not have any boundary condition to impose. To evaluate first degree operators, the number of derivatives in the jumps evaluation, Eq. (8), is reduced,

$$J_\alpha^\pm = \left[u^{(0)}\right]_\alpha + \frac{h^\pm}{1!}\left[u^{(1)}\right]_\alpha + \frac{\left(h^\pm\right)^2}{2!}\left[u^{(2)}\right]_\alpha + \mathcal{O}\left(h^3\right) \quad , \tag{11}$$

and only two primary interpolation points are needed for the wall derivatives:

$$u^{(k)}\left(x_{\alpha^+}\right) = \tilde{S}_\alpha^k u_\alpha + \tilde{S}_1^k u\left(x_{i+2}\right) + \tilde{S}_2^k u\left(x_{i+3}\right) + \mathcal{O}\left(h^{3-k}\right) \quad , \tag{12}$$

for $k = 1, 2$, with $\tilde{S}$, the interpolation coefficients satisfying the needed accuracy. Following the same approach, when no boundary condition is provided, we rather extrapolate the value of the field up to the boundary, leading to another evaluation of the wall derivatives,

$$u^{(k)}\left(x_{\alpha^+}\right) = \tilde{T}_1^k u\left(x_{i+2}\right) + \tilde{T}_2^k u\left(x_{i+3}\right) + \tilde{T}_3^k u\left(x_{i+4}\right) + \mathcal{O}\left(h^{3-k}\right) \quad , \tag{13}$$

with $\tilde{T}$, the extrapolation coefficient satisfying the needed accuracy, $\mathcal{O}\left(h^3\right)$.

The Immersed Interface tool is applicable to a wide variety of boundary conditions. While we only present three of them, the number of combination between the needed accuracy and the kind of boundary condition is quite large.

2.2.2. *Generalization to two dimensions*

When considering 2D applications, we use the dimension splitting approach, i.e. we apply, dimension by dimension, a 1D jump correction to standard 2D stencils. Therefore, while the generalization of a Dirichlet boundary condition is straightforward, imposing a 2D (or, equivalently a 3D) Neumann boundary condition increases the complexity, since the dimensions are linked by the imposed boundary expression and splitted in the IIM. To solve that problem in a coherent way, Marichal [12] proposed to use the so-called *compatible interpolation scheme*. While, in this work, we only consider Dirichlet boundary conditions, the *compatible interpolation scheme* offers a way to compute useful information at the inner interface.

Let us consider a 2D intersection between the body and the mesh, and define the current local direction, ξ, as the direction of the grid line intersecting with the body and the transverse local direction, η, see Fig. 2. Following these definitions and assuming a Dirichlet boundary condition, the derivatives needed in the jump evaluation, Eq. (8), are then expressed, using the notation of Fig. 2 with $\xi_0 < \xi_\alpha \leq \xi_1$, as

$$\left.\frac{\partial^k u}{\partial \xi^k}\right|_\alpha = S_\alpha^{\xi,k}\, u_\alpha + \sum_{i=1}^{3} S_i^{\xi,k}\, u\left(\xi_{i+1}, \eta_0\right) + \mathcal{O}\left(h^{4-k}\right) \tag{14}$$

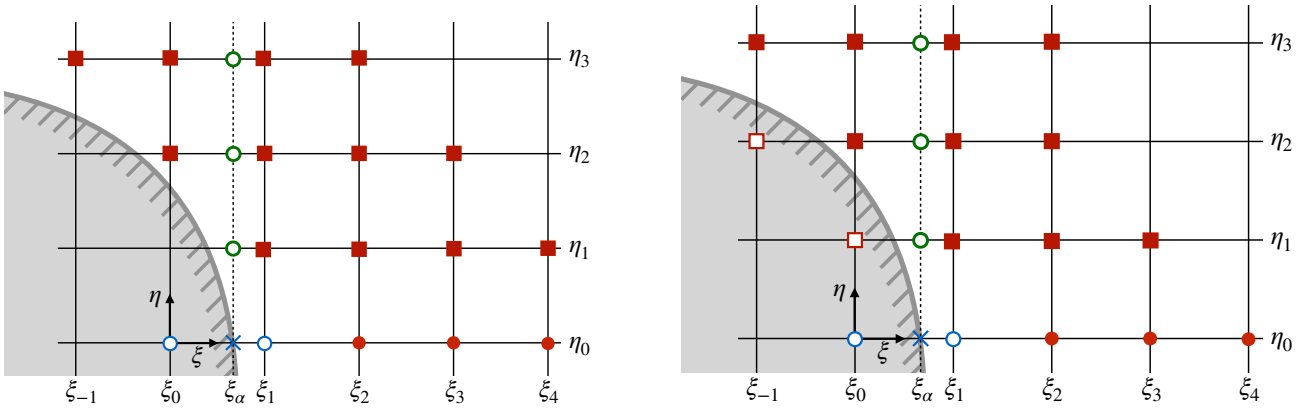
$$\left.\frac{\partial^k u}{\partial \eta^k}\right|_\alpha = S_\alpha^{\eta,k}\, u_\alpha + \sum_{j=1}^{3} S_j^{\eta,k}\, u\left(\xi_\alpha, \eta_j\right) + \mathcal{O}\left(h^{4-k}\right) \tag{15}$$

for $k = 0, ..., 3$ and the missing values in the η direction are interpolated from the neighbours,

$$u\left(\xi_\alpha, \eta_j\right) = \sum_{i=1}^{4} R_i^0 \, u\left(\xi_{i-2+s_{\eta_j}}, \eta_j\right) + \mathcal{O}\left(h^4\right) \quad , \tag{16}$$

where s_{η_j} is a shift variable, computed for each level (see below). Following the notation in 1D, the grid points involved in Eq. (14) and Eq. (15) are called "primary interpolation points", while the points required in Eq. (16) are called "secondary interpolation points". Let us also note that Eq. (14) is a generalization of Eq. (10) and that this formulation is directly transposable to a 3D framework, as in Gillis et al. [24].

The choice of the secondary interpolation points, i.e. the value of the shift coefficient s_{η_i} in Eq. (16), varies from one approach to another. Marichal [12] and Gillis et al. [24] recommend to use the most centred scheme possible to reduce the interpolation error in Eq. (16). However, they discard any point inside the body and shift the interpolation stencil in the flow when it crosses the boundary, see Fig. 2 on the left side. In our work, to improve the consistency at the interface and reduce the number of the secondary interpolation points in the flow, we keep one point in the body, when the secondary interpolation stencil crosses the inner boundary. The value of the inner point is computed by extrapolation in the ξ direction, using the jumps, Eq. (14), at the interface, as it would have been computed when considering primary interpolation in the ξ direction at that level. Following the example in Fig. 2b, the point $u\left(\xi_0, \eta_1\right)$ is computed using Eqs. (8) and (14), evaluated at level η_1. With this choice, we aim at reducing the interpolation error and still consider the best possible information. Still following the example in Fig. 2b, the shift variables are then $s_{\eta_1} = 1$, $s_{\eta_2} = 0$ and $s_{\eta_3} = 0$.



(a) Approach proposed by Marichal [12] and Gillis et al. [24]

(b) Present approach

Figure 2: **2D Immersed interface scheme** - The mesh intersects the interface and generates a control point (×), discontinuity affected points (○) and primary interpolation points in the current direction, ξ, and the transverse direction, η (represented as ● and ●, respectively). Primary interpolation points in the η direction are computed as a polynomial-interpolated value using secondary interpolation points (■ and □).

As already mentioned in 1D, there is a large number of combinations between the needed accuracy and type of boundary condition. We here present two of them: the evaluation of a *first degree* operator and the extrapolation of the field when no information is available from the boundary condition. Following the same approach as in 1D, when evaluating *first degree* operators, we use a degraded version of the presented method: only two primary interpolation points are needed and the wall

derivatives are then expressed as

$$\left.\frac{\partial^k u}{\partial \xi^k}\right|_\alpha = \tilde{S}_\alpha^{\xi,k}\, u_\alpha + \sum_{i=1}^{2} \tilde{S}_i^{\xi,k}\, u\left(\xi_{i+1}, \eta_\alpha\right) + \mathcal{O}\left(h^{3-k}\right) \tag{17}$$

$$\left.\frac{\partial^k u}{\partial \eta^k}\right|_\alpha = \tilde{S}_\alpha^{\eta,k}\, u_\alpha + \sum_{j=1}^{2} \tilde{S}_j^{\eta,k}\, u\left(\xi_\alpha, \eta_j\right) + \mathcal{O}\left(h^{3-k}\right) \tag{18}$$

with

$$u\left(\xi_\alpha, \eta_j\right) = \sum_{i=1}^{3} \tilde{R}_i^0\, u\left(\xi_{i-2+s_{\eta_j}}, \eta_j\right) + \mathcal{O}\left(h^3\right) \quad . \tag{19}$$

Again, s_{η_j} is chosen such that the interpolation error is reduced.

When no information is available from the boundary condition, the field is extrapolated up to the inner wall using the following formula,

$$\left.\frac{\partial^k u}{\partial \xi^k}\right|_\alpha = \sum_{i=1}^{3} \tilde{T}_i^{\xi,k}\, u\left(\xi_{i+1}, \eta_0\right) + \mathcal{O}\left(h^{3-k}\right) \tag{20}$$

$$\left.\frac{\partial^k u}{\partial \eta^k}\right|_\alpha = \sum_{i=1}^{3} \tilde{T}_i^{\eta,k}\, u\left(\xi_\alpha, \eta_i\right) + \mathcal{O}\left(h^{3-k}\right) \tag{21}$$

and Eq. (19). The way we compute the interpolation/extrapolation weights (S, R, T, $\tilde{S}$, $\tilde{R}$, $\tilde{T}$) is detailed in Appendix A along with a detailed description of the Immersed Interface stencils used in Appendix C. Finally, we would like to stress that the construction of those interpolation weights represents a limited computational overhead. Indeed, let us consider the example of a cylinder of radius R and a grid resolution R/h. The surface of the cylinder will cross the grid lines and generate $4R/h$ intersection points per dimension. The interpolation weights are then computed for each of the $16R/h$ biased points generated from the intersection points. The computational overhead, which scales as $\mathcal{O}(R/h)$, is then small compared to the cost of evaluating a source term, $\mathcal{O}\left((R/h)^2\right)$, or to the cost of a Poisson solver, $\mathcal{O}\left((R/h)^2 \log\left((R/h)^2\right)\right)$. For the 2D benchmark cases considered in this work, it represents approximately 5% to 10% of the computational time required by the Poisson solver.

2.3. Time step algorithm of the proposed approach

In this section we present a global view of the successive computations to be performed, for the evaluation of the right hand side of Eq. (3), during one time step. It can also be seen as a sub-step when using multi-step time integration schemes (such as Runge-Kutta schemes).

1. **Poisson - Section 2.4:** Given the bulk vorticity field, ω^n_{bulk}, we compute the no through-flow velocity, $\mathbf{u}^n = \nabla \times \left(\psi^n \mathbf{e}_z\right)$, by solving the Poisson equation

$$\nabla^2 \psi^n = -\omega^n \quad , \tag{22}$$

   such that $\mathbf{u} \cdot \hat{\mathbf{n}}\Big|_{\delta\Omega_b} = 0$, i.e. the no-through-flow condition is satisfied.
2. **Wall vorticity - Section 2.5:** Given $\mathbf{u}^n$, we compute the ω^n_{wall} for each control point, imposing $\mathbf{u}_{wall} = 0$, i.e we compute the vorticity value at the wall in order to impose the no-slip condition.
3. **Diffusion - Section 2.5:** Given ω^n and ω^n_{wall}, we compute the diffusion term, $\nabla^2 \omega^n$ that takes the wall vorticity into account.

4. **Extension - Section 2.6:** Given $\mathbf{u}^n$ and $\nabla^2\omega^n$, we extend those fields into the body using a Hamilton-Jacobi extension.
5. **M2P - Section 2.6:** Given the extended fields, we perform the mesh to particles interpolation of the computed rhs of Eq. (3) and assign a value to the ghost particles.
6. **Advection and diffusion:** We integrate in time the system of Eq. (3) on the particles.
7. **P2M - Section 2.7:** We perform the particle to mesh interpolation (and eventually a redistribution of the particles, if required), and we also erase the spurious extended fields inside the body.

Again, we would like to stress that this sequence actually constitutes an evaluation of the right hand side of Eq. (3). Indeed, this method is not a time splitting algorithm, which consequently allows to reach a global order as high as the consistency of Eq. (3) right hand side discretization, here at third order.

2.4. Poisson equation

The 2D Poisson equation is solved on $\boldsymbol{\psi} = \psi\ \mathbf{e}_z$ using the bulk vorticity $\boldsymbol{\omega}_{bulk} = \omega_{bulk}\ \mathbf{e}_z$ and a no-through flow condition on the body, i.e. a Dirichlet boundary condition,

$$\nabla^2\psi = -\omega_{bulk} \qquad \text{such that} \qquad \psi\Big|_{\delta\Omega_b} = \psi_b \ . \tag{23}$$

In 2D, this is not a simply connected domain and one needs to close the system by imposing an additional constraint. We impose the total circulation of the flow, Γ, which sets the constant ψ_b. To solve the Poisson equation, Eq. (23), we use the efficient immersed interface Poisson solver proposed by Gillis et al. [24]. It decomposes the problem using the Sherman-Morrison-Woodbury formula [25, 26] and actually solves the following equation,

$$\psi = \left(\nabla_h^2 + E_a C E_i^T\right)^{-1} \omega_{bulk} = \nabla_h^{-2}\left\{ I_n - E_a C \left[I_i + E_i^T \nabla_h^{-2} E_a C\right]^{-1} E_i^T \nabla_h^{-2} \right\} \omega_{bulk} \ , \tag{24}$$

where E_a and E_i stand for the domain restriction operators to the discontinuity affected points and primary interpolation points, respectively, I_n is the identity matrix of size n (n is the size of the set of the interpolation points) and C represents the jumps computed using Eq. (14). This kind of approach has been initially proposed in the context of penalization problems by Chatelin and Poncet [27] and is detailed in the context of the Poisson solver in Gillis et al. [24].

The operator $\left[\nabla_h^2\right]^{-1}$ is applied through a FFT-based Poisson solver [35] using the 2D lattice Green's function as a convolution kernel (for more details, see Appendix B, Martinsson and Rodin [36] and Gillman and Martinsson [37]). Let us also observe that the circulation constraint, Γ, is actually imposed through a constraint on the IIM-generated correction, $\left[E_a C E_i^T\right]\psi$, which amounts to a vortex sheet; its strength has to respect $\Gamma_{body} = \Gamma - \Gamma_{bulk}$. The number of iterations required to solve the sub-system is reduced thanks to the use of recycling based on the previous solutions, as proposed by Amritkar et al. [38] and Gillis et al. [17], among others. Finally, the no-through flow enforcing velocity field is obtained by differentiating the stream function and injecting the Dirichlet boundary condition, $\psi = \psi_b$, into Eq. (14).

2.5. Vorticity sheet treatment

The no-through flow enforcing velocity field still has a spurious slip velocity at the wall. This will generate a vorticity sheet, not representable on the grid points since it exists only at the body surface. Following Lighthill [20] approach, this vorticity sheet is the newly created vorticity necessary to satisfy the no-slip condition at the wall and therefore is part of the flow.

A classical way to do so, proposed by Marichal [12] in the framework of IIM and broadly used in vortex methods [20, 18, 19], is to first compute the no-through flow enforcing velocity field and use it to advect particles. The diffusion is performed using a zero-flux boundary condition, hence discarding the generation of vorticity at the wall. One then evaluates the error committed by these two operations, i.e. the spurious residual slip velocity which corresponds to the flux of vorticity that has to enter the flow by another diffusion step. This methodology has two major drawbacks. First, it splits the diffusion operation and therefore bounds the time convergence to $\mathcal{O}(\Delta t)$; second, in order to ensure a high consistency in the slip velocity evaluation, it requires the solution of two Poisson problems.

The present work uses a different technique that merges the two operations previously required into a single diffusion operation, carried out with a Dirichlet boundary condition. We compute the value of the vorticity field at the wall, using the Immersed Interface framework, in order to impose the no-slip behaviour as

$$\omega_{wall} = \frac{\partial u_y}{\partial x}\Big|_{wall} - \frac{\partial u_x}{\partial y}\Big|_{wall} \quad , \tag{25}$$

where the derivatives are computed by injecting $u_x\Big|_{wall} = u_y\Big|_{wall} = 0$ into Eqs. (20) and (21) for each control point on the wall.

The diffusion field, $\nabla^2\omega$, is computed in the fluid domain, and corrected along the boundary by each control point, injecting the value of the wall vorticity, ω_{wall}, into Eq. (14). The proposed method differs significantly from the approach used in BEM [18, 19, 39]: it also diffuses a newly generated vorticity, obtained from the no-slip condition, into the flow; yet, it uses a Dirichlet boundary condition to obtain ω_{wall}, and does the diffusion step using that value, together with the convection step. This is as opposed to doing the diffusion step with zero flux, together with the convection step, and then, afterwards, computing the vorticity flux, using a Neumann boundary condition, and diffusing it into the flow (see Marichal [12] for a stability comparison of both approaches).

2.6. Mesh to Particle (M2P) interpolation and extension inside the body

The mesh to particles interpolations use the M_4' interpolation kernel [30, 31], which is equivalent to a stencil of width $4h$. In the neighbourhood of the interface, one could use a decentered interpolation in order to use only the information in the flow; or use extended field values to reproduce the effect of a decentered scheme. To keep the interpolation process unchanged, we rather choose the second option by extending the field inside the body, as initially proposed by Marichal et al. [23]. For this work, we identified two different extensions of the considered field inside the body, which generalizes the work proposed by Aslam [40] to the Immersed Interface framework. For the sake of completeness, we present both approaches below. Both approaches consider the body level-set function, ϕ, i.e. a signed distance function such that $\phi < 0$ describes the region inside the body. For each point inside the body, we define the inner normal

$$\hat{\mathbf{n}}_{in} = -\frac{\nabla\phi}{\left|\nabla\phi\right|} \quad , \tag{26}$$

and, in order to extend the field in a small band around the body interface only (since we only need a few ghost points inside the body), the cutting function, $\mathcal{C}$,

$$\mathcal{C} = \begin{cases} 0 & \text{if} \quad \phi \geq 0 \\ \dfrac{1}{1 + \exp\left(10^3 \dfrac{\left|\phi - h_c\right|}{L}\right)} & \text{if} \quad \phi < 0 \end{cases} , \tag{27}$$

where L is the characteristic length of the body and h_c is the cutting distance. The extrapolated field, f, is obtained as the stationary solution of the following system

$$\frac{\partial}{\partial \tau}\begin{bmatrix} f \\ f' \end{bmatrix} + \mathcal{C}\begin{bmatrix} \hat{\mathbf{n}}_{in} \cdot \nabla & -1 \\ 0 & \hat{\mathbf{n}}_{in} \cdot \nabla \end{bmatrix}\begin{bmatrix} f \\ f' \end{bmatrix} = 0 , \tag{28}$$

where the unknowns are extended with the normal derivative of f, $f' \triangleq \hat{\mathbf{n}}_{in} \cdot \nabla f$. This system is solved in pseudo time τ until convergence of the solution, using an up-wind scheme coupled with a RK2 integration.

The two identified approaches differs in the way they take the boundary into account and in the way they compute the normal derivative. The first approach decomposes the problem of the extrapolation into two steps. First, knowing a field f outside the body, we compute its value and its normal derivative, $f' \triangleq \hat{\mathbf{n}}_{in} \cdot \nabla f$, at the boundary using Eq. (14) and Eq. (15), as in [23]. Then, once all the needed information is known at the boundary, we extend it from the boundary into the body using a Hamilton-Jacobi system, coupled with the immersed interface technique and the pre-computed surface information to evaluate the right hand side of Eq. (28). This approach has two drawbacks: first, one needs to use the IIM to correct the evaluation at each iteration and second, in order to use a high order up-wind discretization of the gradient, more than one grid point on each side of the interface need to be affected by the Immersed Interface correction.

The second approach is much simpler, as we do not have to take the interface explicitly into account while integrating the system Eq. (28). Indeed, before integrating the Hamilton-Jacobi system, we pre-compute f and $f' = \hat{\mathbf{n}}_{in} \cdot \nabla f$ in the fluid-side vicinity of the wall, using centered finite differences, hence second order, and correcting the discontinuity affected points with the Immersed Interface jumps. Since we do not want to consider any boundary condition for the extended field, we extrapolate the value and the derivatives (using third order extrapolation) up to the boundary, using Eq. (20) and Eq. (21), which is in agreement with the global accuracy of the scheme, $\mathcal{O}\left(h^2\right)$. This way, we build a second order evaluation of f' and still keep the number of discontinuity affected points to one, on each side of the interface.

In this work, we chose to use a linear extension in the normal direction. A higher degree extension (e.g. using $\frac{\partial}{\partial \tau} f''$ in Eq. (28)) would lead to an increased accuracy; but it would also require higher order polynomials for the extrapolation, which become poorly conditioned when considering the extension of stiff fields. Nevertheless, the proposed method generalizes itself easily to higher extension degrees.

The proposed Hamilton-Jacobi extension is then used on the velocity and vorticity fields to extend them into the body and provide enough grid-based information for a M2P operation near the boundary.

2.7. P2M and ghost particles

The Particle-to-Mesh (P2M) operation follows the same approach as for the M2P operations. Ghost particles are constructed in order to provide a full particle support for the standard P2M interpolation kernels at grid points close to the interface. The way to create a ghost particle relies on the following observation: the field that needs to be interpolated is the diffused strength of the particles and there is no easy and cheap way to create ghost particles from the flow particle status. Therefore, we propose to build ghost particles using the same extension of the vorticity inside the body, during the M2P operation. Ghost particles are then advected and diffused along with the flow particles, using the extension of the velocity, vorticity and diffusion fields (see Section 2.6).

3. Validation

In order to validate the spatial and temporal convergences of the proposed method, we consider the diffusion of a Gaussian vorticity field, centered at $\mathbf{x} = \mathbf{x}_c$:

$$\omega_G(x, y, t) = \frac{\Gamma}{\pi} \frac{1}{\left(\sigma^2 + 4\nu t\right)} \exp\left(-\frac{r^2}{\left(\sigma^2 + 4\nu t\right)}\right) \quad , \tag{29}$$

with $r = \left|\mathbf{x} - \mathbf{x}_c\right|$. The velocity, $\mathbf{u} = (u\ ,\ v)$, and diffusion, $\nabla^2\omega$, are analytically obtained:

$$u = -\frac{\Gamma}{2\pi}\frac{y}{r^2}\left[1 - \exp\left(-\frac{r^2}{\left(\sigma^2 + 4\nu t\right)}\right)\right] \tag{30}$$

$$v = \frac{\Gamma}{2\pi}\frac{x}{r^2}\left[1 - \exp\left(-\frac{r^2}{\left(\sigma^2 + 4\nu t\right)}\right)\right] \tag{31}$$

$$u_\theta = \frac{\Gamma}{2\pi r}\left[1 - \exp\left(-\frac{r^2}{\left(\sigma^2 + 4\nu t\right)}\right)\right] \tag{32}$$

$$\nabla^2\omega = \frac{4\Gamma}{2\pi}\frac{1}{\sigma^4}\left[\frac{r^2}{\sigma^2} - 1\right]\exp\left(-\frac{r^2}{\left(\sigma^2 + 4\nu t\right)}\right) \tag{33}$$

To validate the proposed method, we clip that Gaussian vorticity field, by removing its core, $r < R = \sigma$, and we replace it with a rotating cylinder. The flow lies outside, $r \geq R = \sigma$, and has a vorticity field given by $\omega = \omega_G$. The removed core effect is ensured by the adequate rotating velocity of the cylinder, i.e. the one that reproduces that of the diffusing Gaussian vortex at $r = R$: $u_{\theta\ wall} = u_\theta(r = R)$. The corresponding circulation of the body is then

$$\Gamma_{body} = \Gamma\left[1 - \exp\left(-\frac{R^2}{\left(\sigma^2 + 4\nu t\right)}\right)\right] \quad . \tag{34}$$

The analytical solution of the diffusing Gaussian vorticity thus remains unchanged and is still valid in the fluid domain. The Reynolds number is defined using the characteristic velocity, $U = \frac{\Gamma}{2\pi R}$, and the diameter, $D = 2R$, as

$$\mathrm{Re} = \frac{U\ D}{\nu} = \frac{1}{\pi}\ \frac{\Gamma}{\nu} \quad . \tag{35}$$

Finally, we also define the dimensionless time, $t^* = t\ \Gamma/R^2$.

We run three different validation tests on this benchmark, using the analytical expression of the vorticity given by Eq. (29), as an initial condition, and the analytical wall velocity given by Eq. (30) and Eq. (31) at every time. The first two results prove the third order time convergence of the diffusion alone and of the full method (without remesh), respectively; the last one shows the second order spatial convergence of the full method (without remesh). For every run, we use a very narrow tolerance for the Poisson iteration, 10^{-14}, 6 vectors in the recycling subspace and $\mathrm{Re} = 100$.

We study the convergence of the 2- and ∞-norms of the error. Given a field of reference, the dimensionless norms are defined as

$$E_2 = \frac{R}{\Gamma}\sqrt{\sum_{i,j}\left(\omega_{ij}^{ref} - \omega_{ij}\right)^2 \; h^2} \qquad\qquad E_\infty = \frac{R^2}{\Gamma}\max_{ij}\left|\omega_{ij}^{ref} - \omega_{ij}\right| \; . \tag{36}$$

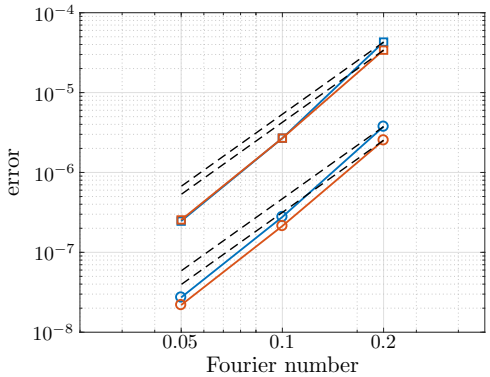
Considering two successive resolutions, (Δ_k) and (Δ_{k+1}), and their associated errors, we define the local convergence rate, r_2 and r_∞, as

$$r_2 = \frac{\log\left[E_2^k/E_2^{k+1}\right]}{\log\left[\Delta_k/\Delta_{k+1}\right]} \qquad\qquad r_\infty = \frac{\log\left[E_\infty^k/E_\infty^{k+1}\right]}{\log\left[\Delta_k/\Delta_{k+1}\right]} \; . \tag{37}$$

3.1. Temporal convergence

3.1.1. Diffusion

The diffusion problem presented earlier is purely axisymmetric and hence we first do not advect the particles, in order to highlight the diffusion treatment presented in Section 2.3. Other operations of the proposed time-step remain unchanged. This special treatment for the advection avoids the errors produced by the particle-mesh interpolation and particle distortion. The runs are made using $R/h = 51.2$ (i.e. a grid of 512×512 for a domain of $10R \times 10R$) and for three Fourier numbers: $r \triangleq \nu\Delta t/h^2 = \{0.2\ ;\ 0.1\ ;\ 0.05\}$. We use these runs in a self-convergence test, as the error is computed with respect to a reference solution computed very accurately, using a much smaller Fourier number $r = 0.005$. Indeed, the spatial discretization produces an error much higher than the time integration and, therefore, does not allow to compare to the analytical vorticity solution. We compare the obtained results at two dimensionless times, $T_1^* = 0.2\,\pi\,\mathrm{Re}\,(h/R)^2 \simeq 0.024$ and $T_2^* = 4T_1^* \simeq 0.096$. The results of the convergence are presented in Fig. 3 and exhibit a third order convergence, which validates the proposed



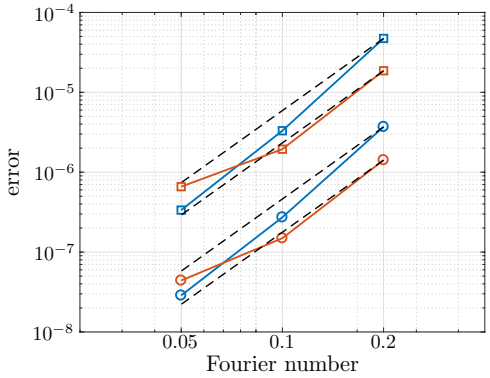
T_1^*					
Fourier number	time step	E_∞	r_∞	E_2	r_2
0.2	1	4.29e-05		3.75e-06	
0.1	2	2.69e-06	4.0	2.77e-07	3.8
0.05	4	2.46e-07	3.5	2.73e-08	3.3
T_2^*					
Fourier number	time step	E_∞	r_∞	E_2	r_2
0.2	4	3.40e-05		2.53e-06	
0.1	8	2.69e-06	3.7	2.14e-07	3.6
0.05	16	2.55e-07	3.4	2.19e-08	3.3

Figure 3: **Temporal convergence** - diffusion: E_∞ (T_1^* : and T_2^* :), E_2 (T_1^* : and T_2^* :) and $\mathcal{O}\left(\Delta t^3\right)$ (- - ·).

approach for the diffusion. Indeed, the treatment of the diffusion term on the particles does not include any splitting and hence does not affect the third order Runge-Kutta integration scheme.

3.1.2. *Diffusion and advection*

In this section, we add the advection of the particles in order to highlight the influence of the particle-mesh interpolation and the particles distortion on the accuracy of the method. We then use every sub-step of the algorithm described in Section 2.3. However, we do not remesh the particles during the simulation. We use a time-step fixed by the Fourier number, $r \triangleq \nu \Delta t / h^2 = \{0.2\ ;\ 0.1\ ;\ 0.05\}$ and the same mesh resolution as in the previous section. Again, this analysis is based on a self-convergence approach as the spatial discretization produces much higher errors than the time integration. The results are presented in Fig. 4. The third order convergence is reached for T_1^*. The change in convergence rate observed for T_2^* is mainly due to the lagrangian deformation that begins to dominate. It could also be partially attributed to the non-linear behavior of the extension methodology, which depends on both Δx and Δt and hence deteriorates the self-convergence order.

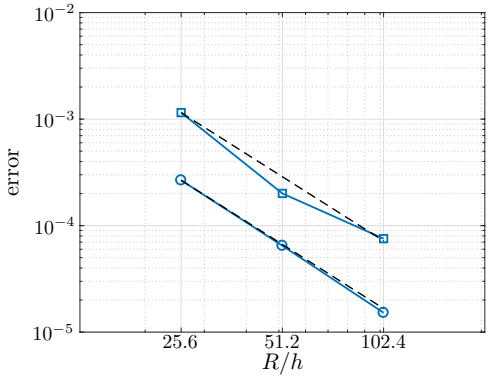


T_1^*					
Fourier number	time step	E_∞	r_∞	E_2	r_2
0.2	1	4.73e-05		3.70e-06	
0.1	2	3.30e-06	3.8	2.72e-07	3.8
0.05	4	3.35e-07	3.3	2.86e-08	3.2
T_2^*					
Fourier number	time step	E_∞	r_∞	E_2	r_2
0.2	4	1.86e-05		1.42e-06	
0.1	8	1.94e-06	3.3	1.49e-07	3.2
0.05	16	6.58e-07	1.6	4.41e-08	1.8

Figure 4: **Temporal convergence** - advection and diffusion: E_∞ (T_1^* : and T_2^* :), E_2 (T_1^* : and T_2^* :) and $\mathcal{O}\left(\Delta t^3\right)$ (- - ·).

3.2. *Spatial convergence of the method*

To study the spatial convergence of the method (without remesh), we use the same Gaussian benchmark and consider this time the convergence to the analytical solution. The convergence is run at constant Fourier number, $r = 0.15$ and using three different grid sizes $R/h = [25.6\ ;\ 51.2\ ;\ 102.4]$. We measure the convergence at $T^* = 0.6\ \pi\, \mathrm{Re}\left(h_0/R\right)^2 \simeq 0.29$, when the lagrangian deformation is still acceptable and no remeshing operation is required. The results are shown in Fig. 5 and they present the expected second order convergence for both the 2- and ∞-norms.



T^*					
R/h	time step	E_∞	r_∞	E_2	r_2
25.6	4	1.15e-03		2.66e-04	
51.2	16	2.00e-04	2.5	6.47e-05	2.0
102.4	64	7.54e-05	1.4	1.52e-05	2.1

Figure 5: **Spatial convergence** - E_∞ ( ), E_2 ( ) and $\mathcal{O}\left(h^2\right)$ (- - ·).

4. The impulsively started flow past a cylinder

To illustrate the presented approach, we present its application to the benchmark of the impulsively started flow past a cylinder, and we compare our results to several published works. We perform the comparison between the grid resolutions of several runs by quantifying the resolution in the boundary layer. Therefore, the scaling concerning the spatial resolution h relatively to the characteristic boundary layer thickness δ is here defined as

$$N_\delta \triangleq \frac{1}{\sqrt{\mathrm{Re}_D}} \left(\frac{D}{h}\right) \quad , \tag{38}$$

with $\mathrm{Re}_D = \frac{U_\infty D}{\nu}$ and $\delta \propto \frac{D}{\sqrt{\mathrm{Re}_D}}$, for the flow near the stagnation point, as also used in Mimeau et al. [15]. Moreover, the accuracy of numerical methods in regions with sharp gradients is driven by the mesh Reynolds number based on the vorticity:

$$\mathrm{Re}_{\omega,h} = \frac{\left|\omega\right| h^2}{\nu} \quad . \tag{39}$$

So, a second "figure of merit" to assess the quality, within the boundary layer, of simulations relatively to each other can be defined as

$$Q \triangleq \left(\frac{U_\infty}{D}\sqrt{\mathrm{Re}_D}\right)\frac{h^2}{\nu} = \frac{\sqrt{\mathrm{Re}_D}}{N_\delta^2} = \mathrm{Re}_D^{3/2}\left(\frac{h}{D}\right)^2 \quad , \tag{40}$$

where we have used that the wall vorticity is proportional to $\frac{U_\infty}{\delta}$, thus $\omega_{wall} \propto \frac{U_\infty}{D}\sqrt{\mathrm{Re}_D}$. We also define the following dimensionless quantities,

$$t^* = \frac{t\,U_\infty}{D} \qquad C_D = \frac{F_x}{\frac{1}{2}\,D\,U_\infty^2} \qquad \omega^* = \frac{\omega\,D}{U_\infty} \quad , \tag{41}$$

where F_x is the drag computed using a control volume approach by Noca [41], as also proposed by Mimeau et al. [15],

$$\mathbf{F} = -\frac{d}{dt}\int_{\Omega_c} \mathbf{u}\, d\mathbf{x} + \int_{\partial\Omega_c} \boldsymbol{\gamma}\cdot\hat{\mathbf{n}}\, d\mathbf{x} \quad , \tag{42}$$

where Ω_c is the control volume, $\hat{\mathbf{n}}$ is normal to $\delta\Omega_c$ and

$$\begin{aligned}\boldsymbol{\gamma} = \frac{1}{2}\left|\mathbf{u}\right|^2\mathbf{I} \quad & -\frac{1}{N-1}\mathbf{u}\,(\mathbf{x}\times\boldsymbol{\omega}) + \frac{1}{N-1}\boldsymbol{\omega}\,(\mathbf{x}\times\mathbf{u}) \\ & -\frac{1}{N-1}\left[\left(\mathbf{x}\cdot\frac{\partial\mathbf{u}}{\partial t}\right)\mathbf{I} - \mathbf{x}\frac{\partial\mathbf{u}}{\partial t}\right] \\ & +\frac{1}{N-1}\left[\left[\mathbf{x}\cdot(\nabla\cdot\mathbf{T})\right]\mathbf{I} - \mathbf{x}\,(\nabla\cdot\mathbf{T})\right] + \mathbf{T} \quad .\end{aligned} \tag{43}$$

in which N is the spatial dimension (2 or 3), $\mathbf{I}$ is the unit tensor and $\mathbf{T} = \mu\left[\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right]$ is the viscous stress tensor. This approach is well suited for the vorticity formulation as it does not involve the pressure field. Finally, we define $\theta \in [-\pi\ ;\ \pi[$, the angle along the cylinder surface such that the stagnation point is at $\theta = 0$.

To alleviate the computational cost, we use a subdomain, $\left[N_x^{sub} \times N_y^{sub}\right]$ that fully includes the immersed body and where the Poisson sub-system is solved with a relative tolerance of 10^{-6} and 6 recycling solutions. The cylinder is centered at $(0.75D\ ;\ 1.25\ D)$ and we use the following accuracy parameters: $C_{\mathrm{LCFL,S}} = 0.1$, $C_{\mathrm{LCFL},\omega} = 0.2$ and $r = 0.3$. The remaining parameters are presented in Table 1.

Re	R/h	N_δ	Q	L_x	L_y
550	102.4	8.73	0.31	5 D	2.5 D
3000	204.8	7.48	0.98	5 D	2.5 D
9500	819.2	16.81	0.34	2.5 D	2.5 D
9500	409.6	8.40	1.38	5 D	2.5 D
40000	819.2	8.19	2.98	2.5 D	2.5 D

Table 1: **Impulsively started flow past a cylinder** - setup used for different Reynolds numbers. The computational domain spans $\left[L_x\ ;\ L_y\right]$.

4.1. The impulsively started flow past a cylinder at Re = 550

The flow around a cylinder at Re = 550 is presented in Fig. 8. The simulation parameters are presented in Table 1. The obtained drag coefficient is shown in Fig. 6, where we match the results already obtained by Koumoutsakos and Leonard [18] and Marichal [12]. We also compare favorably to the theoretical drag curve [42], valid only for short times,

$$C_D = 4\sqrt{\frac{\pi}{\left(\mathrm{Re}_D\ t^*\right)}} + 2\pi\left(9 - \frac{15}{\sqrt{\pi}}\right)\frac{1}{\mathrm{Re}_D}\quad . \tag{44}$$

In Fig. 7, we compare the vorticity at the wall used in our method with the one obtained by Koumoutsakos and Leonard [18] and the one provided by Marichal [12]. We observe a behavior consistent with the sharp treatment of the immersed interface techniques. We are indeed closer to the results of Marichal [12] than to those of Koumoutsakos and Leonard [18]. Nevertheless, we predict slightly higher wall vorticity values for $\theta \simeq \pm 0.9\pi \simeq \pm 80°$ compared to the former. Finally, the small oscillations present in the wall vorticity are due to the varying distance between the surface and the first point in the bulk in the X and Y directions: this distance governs the computation of the wall quantities, as described in Section 2.5. These oscillations tend to disappear at higher resolutions, as they are driven by the Q quality criterion, see Figs. 7, 10 and 13.

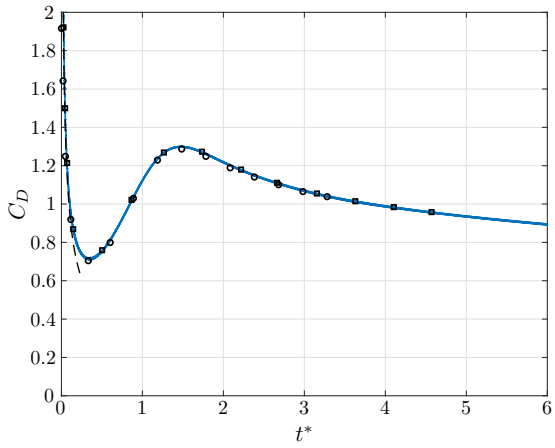


Figure 6: **Drag coefficient at $\mathrm{Re}_D = 550$** - Present approach (—●—), theoretical drag curve (– – ·), Koumoutsakos and Leonard [18] (□) and Marichal [12] (○)

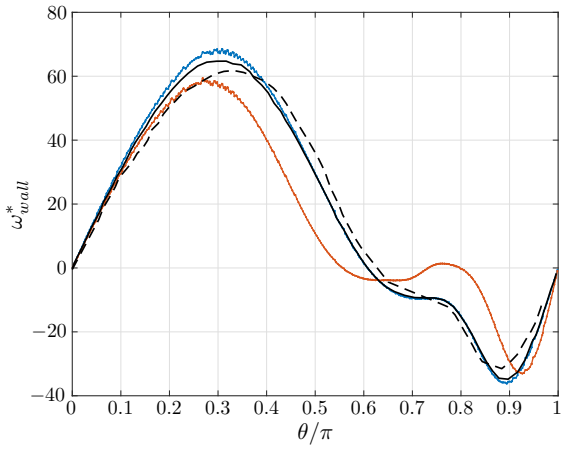
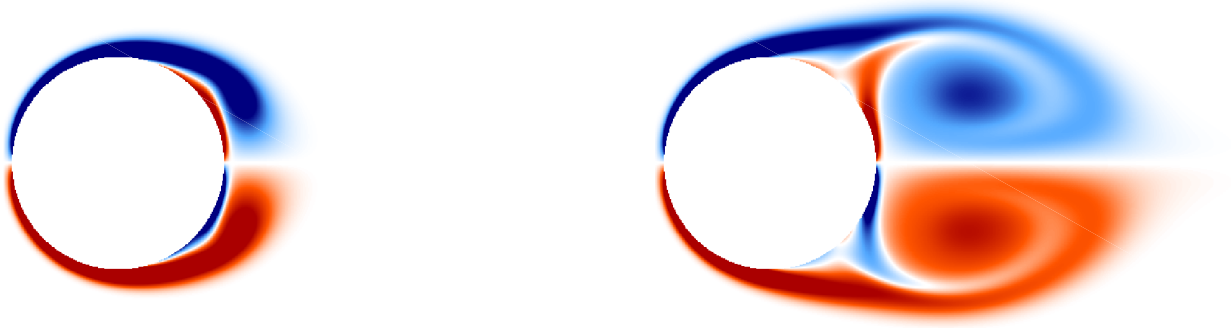


Figure 7: **Wall vorticity at $\mathrm{Re}_D = 550$** - Present approach ($t^* = 1$: —●—, $t^* = 3$: —●—), Koumoutsakos and Leonard [18] ($t^* = 1$: – – ·) and Marichal [12] ($t^* = 1$: ——).

4.2. The impulsively started flow past a cylinder at Re = 3000

The flow around a cylinder at $\mathrm{Re}_D = 3000$ is presented in Fig. 11. The simulation parameters are presented in Table 1. In Fig. 9, we compare the obtained drag coefficient with the results of Koumoutsakos and Leonard [18]. The agreement with



(a) $t^* = 1$ (b) $t^* = 4$

Figure 8: **Vorticity contours at $\mathrm{Re}_D = 550$**

the reference is excellent, in particular the capture of the peak around $t^* \simeq 2$. The dimensionless wall vorticity is exposed in Fig. 10 and compared with the results from Koumoutsakos and Leonard [18]. It can be seen to have more slight oscillations than at $\mathrm{Re}_D = 550$, which are due to the fact that we kept N_δ constant from one run to another, hence decreasing the accuracy at the wall (as we increase the value of Q). Moreover, we note that our approach predicts higher values around wall vorticity peaks than the latest reference; this reflects the better sharpness of the IIVPM method at the wall.

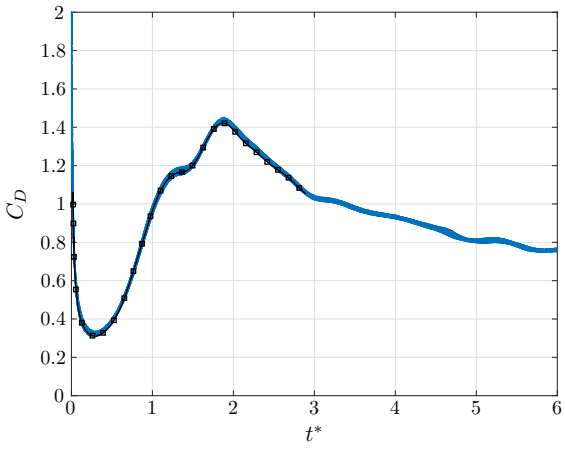


Figure 9: **Drag coefficient at $\mathrm{Re}_D = 3000$** - Present approach (—●—) and Koumoutsakos and Leonard [18] (—□—)

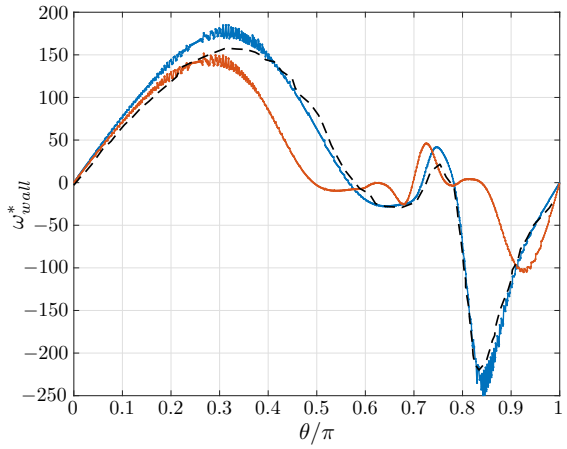
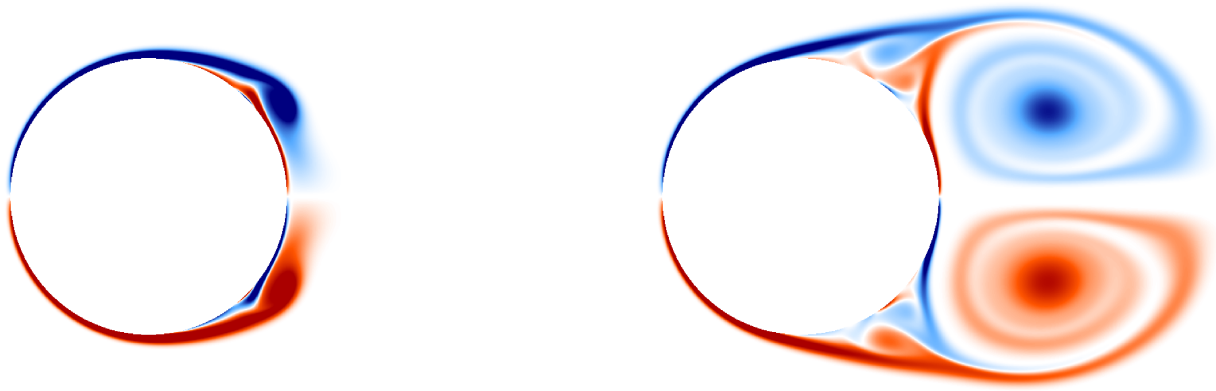


Figure 10: **Wall vorticity at $\mathrm{Re}_D = 3000$** - Present approach ($t^* = 1$: —●— , $t^* = 3$: —●—) and Koumoutsakos and Leonard [18] ($t^* = 1$: - - ·)

4.3. The impulsively started flow past a cylinder at $\mathrm{Re} = 9500$

The flow around a cylinder at $\mathrm{Re}_D = 9500$ is presented in Fig. 14. For this test case, we have used two resolutions to highlight the accuracy of the IIVPM approach and the influence of the accuracy parameter, Q. The coarser computation has been performed with $N_\delta \simeq 8$, while the finer used twice that resolution, $N_\delta \simeq 16$ (i.e. $R/h \simeq 410$ and $R/h \simeq 820$ respectively, see Table 1).

The obtained drag coefficient is shown in Fig. 12, where we observe excellent agreement between the two resolutions. It proves that, for a value of $N_h = 8.4$ (and $Q = 1.38$), the drag coefficient is already quite converged, even when considering high Reynolds number flows. This coefficient can also be seen to globally agree with the results of Rossinelli et al. [43],



(a) $t^* = 1$ (b) $t^* = 4$

Figure 11: **Vorticity contours at $\mathrm{Re}_D = 3000$**

obtained using an effective number of points per radius 8 times higher ($R/h \simeq 3280$ and $N_\delta \simeq 64$ in their case). The more noticeable differences lie in the drag oscillations around $t^* \simeq 2$, which appear to be sharply captured by the present method, even at low resolution. This observation, together with the convergence test of the previous sections, lead us to the claim that the differences with Rossinelli et al. [43], in particular for the dynamics of the shear layer, is due to the dissipation/damping behavior of the penalization method[1]. This in turn influences the generation of the structures shed in the wake and has a noticeable impact on the drag coefficient.

Finally, Fig. 13 shows the wall vorticity for both resolutions. Again we observe higher values around the peak compared to Koumoutsakos and Leonard [18] at $t^* = 1$, which is also observable in the drag coefficient behaviour. Furthermore, the influence of the wall accuracy parameter, Q, is noticeable; increasing the accuracy at the wall leads to significantly smoother wall vorticity values, as expected.

4.4. The impulsively started flow past a cylinder at $\mathrm{Re} = 40000$

In order to stress our approach, the flow around a cylinder at $\mathrm{Re}_D = 40000$ is presented in Fig. 17. We decided to keep the scale N_δ constant to a value comparable to the one used in the previous runs, therefore degrading even more the wall accuracy ($Q \simeq 3$). The other simulation parameters are presented in Table 1.

The obtained drag coefficient is shown in Fig. 15 and compared to that of Rossinelli et al. [43]. We note a qualitative agreement with this reference up to $t^* \simeq 1.2$, which corresponds to the detachement of the vorticity patch located at $\theta \simeq \pm 135°$ in Fig. 17c. A zoom of the vorticity field, shown in Fig. 18, focuses on the region of successive separations; the already detached shear layer exhibits azimuthal oscillations which lead to vortex formations, more intense pressure gradients and eventually more separations. Its behaviour is extremely sensitive to resolution effects (N_δ is 16 times higher in their results than in ours), to the accuracy treatment at the wall (Q) and to the performance of the numerical scheme, which explains the observed difference. Finally, to stress the importance of the wall accuracy criterion, Q, we present the wall vorticity in Fig. 16. It can be observed to present oscillations especially in regions of vortex formation.

[1] There is a factor 2 in the dimensionless time between [43] and the present work since we have used the diameter and they have used the radius in the time

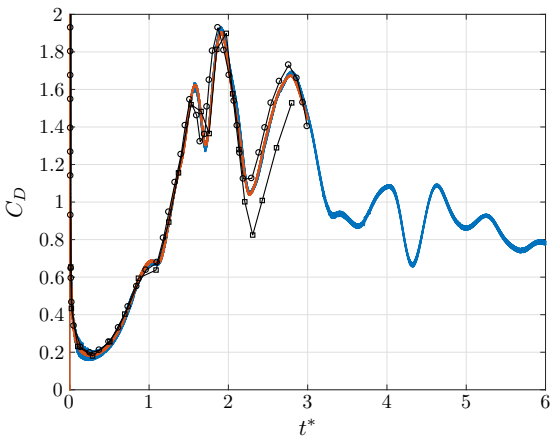


Figure 12: **Drag coefficient at $\mathbf{Re_D = 9500}$** - Present approach ($N_\delta = 8.4$: —●— , $N_\delta = 16.8$: —●—), Koumoutsakos and Leonard [18] (—□—) and Rossinelli et al. [43] (—○—)

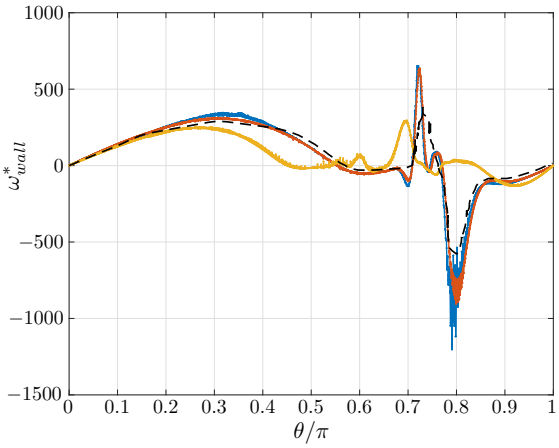
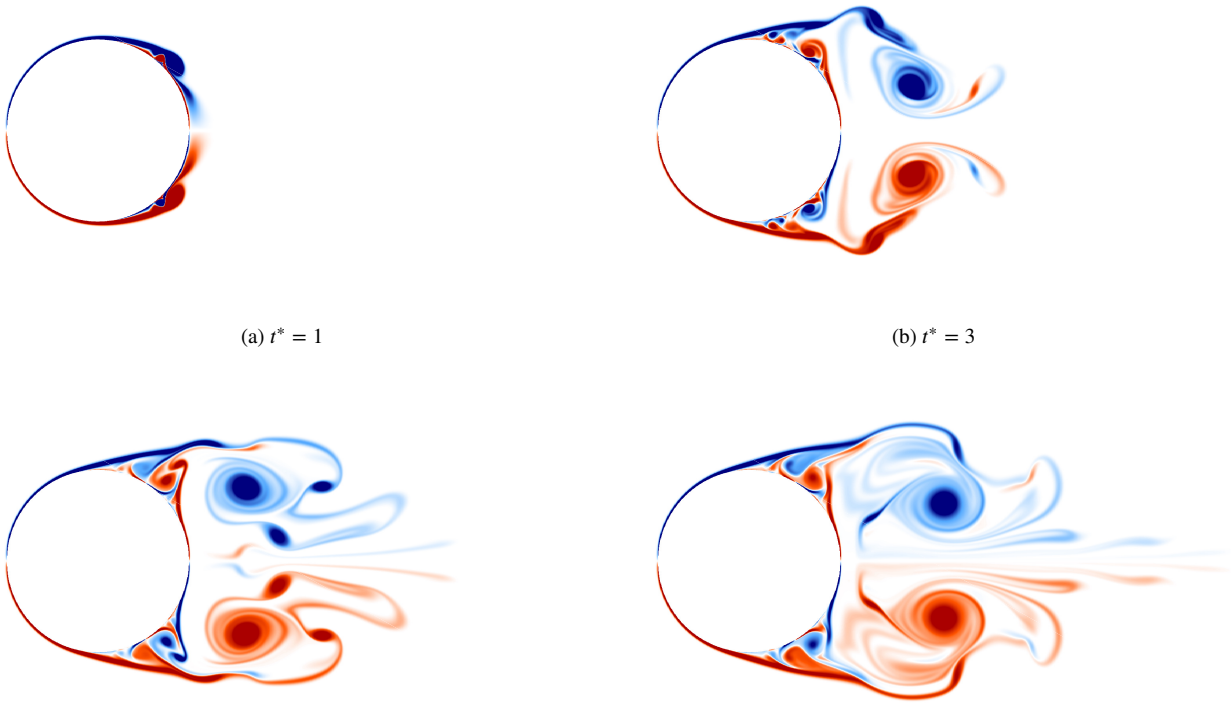


Figure 13: **Wall vorticity at $\mathbf{Re_D = 9500}$** - Present approach at $t^* = 1$ ($N_\delta = 8.4$: —●— , $N_\delta = 16.8$: —●—); at $t^* = 3$ ($N_\delta = 16.81$: —●—) and Koumoutsakos and Leonard [18] at $t^* = 1$ (- - ·)



(a) $t^* = 1$

(b) $t^* = 3$

(c) $t^* = 4$

(d) $t^* = 5$

Figure 14: **Vorticity contours at $\mathbf{Re_D = 9500}$**

5. Conclusions

In this work, we presented a sharp and accurate approach to take an inner obstacle into account in the Vortex Particle-Mesh (VPM) framework. The proposed method relies on the Immersed Interface tool to handle interface discontinuities on

definition.

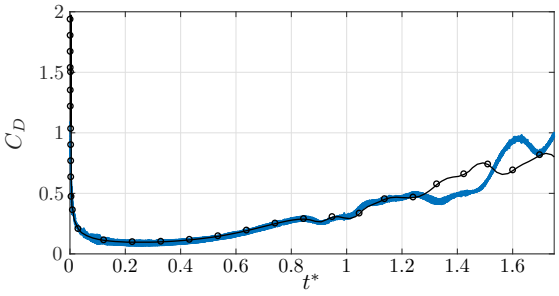


Figure 15: **Drag coefficient at $\mathbf{Re_D = 40000}$** - The present approach ( ) and Rossinelli et al. [43] ( )

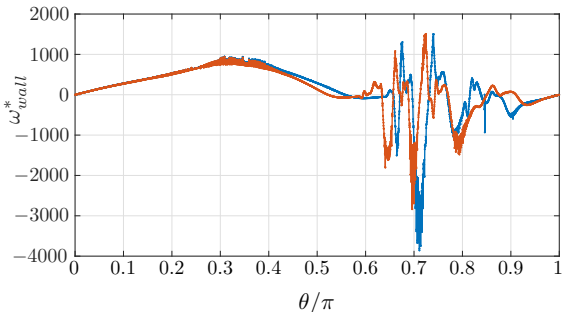
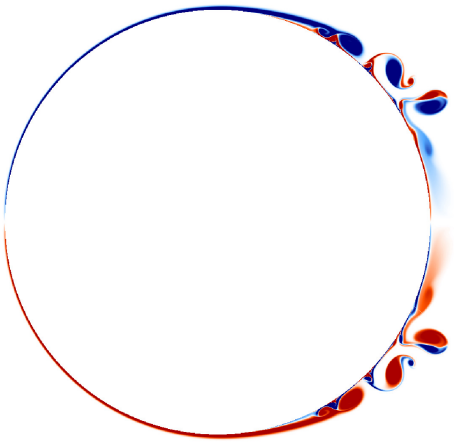
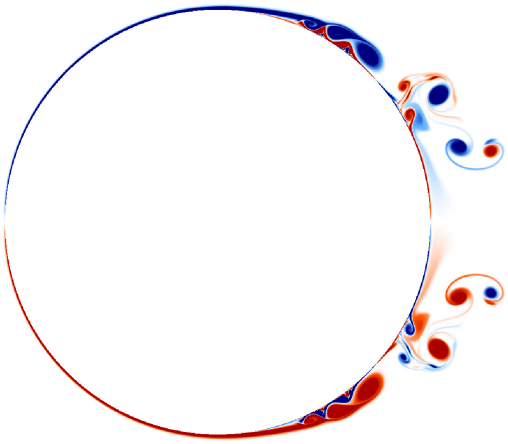


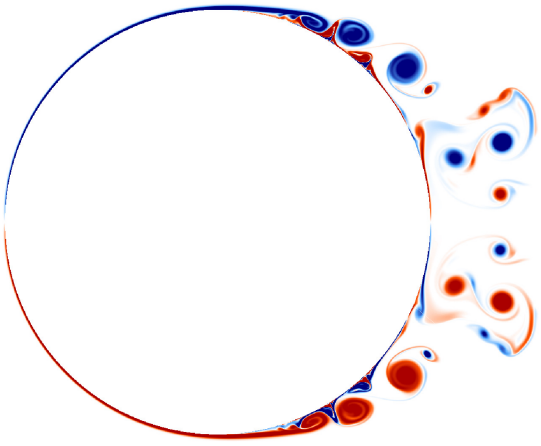
Figure 16: **Wall vorticity at $\mathbf{Re_D = 40000}$** - The present approach ($t^* = 1.0$: , $t^* = 1.5$:)



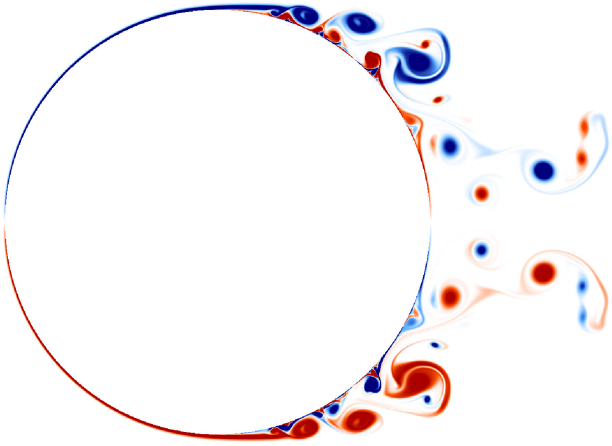
(a) $t^* = 1$



(b) $t^* = 1.2$



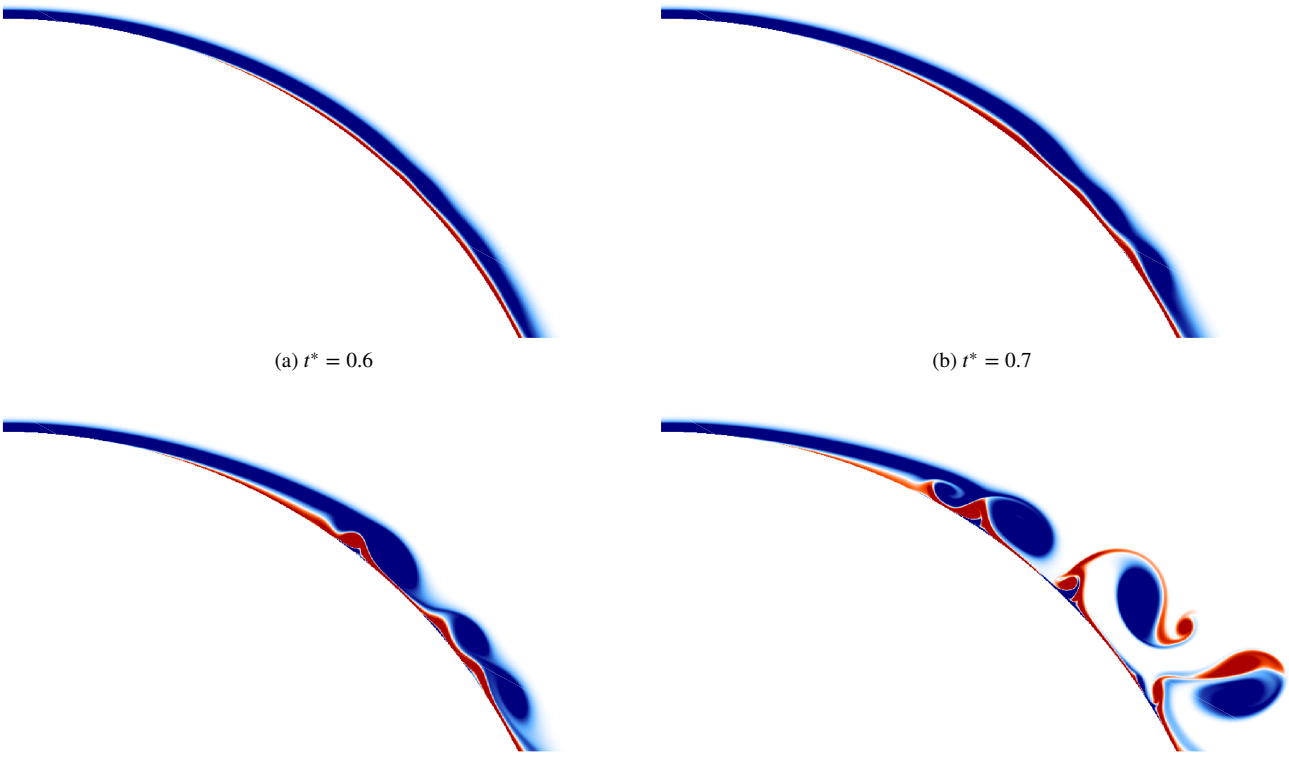
(c) $t^* = 1.5$



(d) $t^* = 1.75$

Figure 17: **Vorticity contours at $\mathbf{Re_D = 40000}$**

the grid and uses ghost particles to impose the body interface from the particle point of view. Every step of the standard VPM methodology is modified using the Immersed Interface approach: the Poisson equation is solved using the decomposition into



(a) $t^* = 0.6$ (b) $t^* = 0.7$

(c) $t^* = 0.8$ (d) $t^* = 1.0$

Figure 18: **Vorticity contours** $\mathrm{Re}_D = 40000$

a global and a local problem, as proposed by Gillis et al. [24], while the diffusion is computed using IIM-modified stencils along with a Dirichlet boundary condition, computed from the no-slip condition.The diffusion and vorticity fields are then extrapolated inside the body, using an IIM-modified Hamilton-Jacobi extrapolation, in order to ensure the accuracy of the mesh-particle interpolation. After advection and diffusion on the particles (including the ghost particles in the body), we interpolate them back onto the mesh. A remeshing technique is also implemented and applied whenever required.

We prove the second order convergence of the proposed methodology, up to the wall, and the conservation of the temporal convergence order, unlike any already proposed immersed interface approach in the VPM framework. Moreover, the present approach only entails one Poisson equation per substep, unlike a flux-based (or Lighthill) approach, which further reduces the computational cost. We assess the computational efficiency on the impulsively started cylinder benchmark for various Reynolds numbers (from $\mathrm{Re} = 550$ to $\mathrm{Re} = 40000$) and compare favorably with other references. We also introduce a “figure of merit" of immersed boundary methods, Q, that quantifies the accuracy of treatment of an interface, and in particular the computation of the wall vorticity. We illustrate that scaling the grid resolution with $\mathrm{Re}^{1/2}$ is not sufficient to insure an equally accurate treatment of the boundary, thus a scaling with $\mathrm{Re}^{3/4}$ would be more appropriate, keeping a constant Q (typically $Q \leq 1$). We finally show that the present IIVPM method is able to reproduce results from Rossinelli et al. [43], at high Reynolds number ($\mathrm{Re} = 9500$), using an eight times coarser grid resolution at the boundary, hence highlighting the gain of accuracy brought by the sharp interface treatment.

This work, so far, has been focused on a second-order accurate method, in the bulk and at the wall. However, it could readily be extended to higher order by enlarging the width of the primary and secondary interpolation layers around the interface.

Moreover, the present approach is extendable to 3D without significant difficulties; this has been already achieved for the Poisson solver by Gillis et al. [24].

Acknowledgements

We would like to acknowledge the insightfull discussions with M. Duponcheel (UCLouvain, Belgium). T. Gillis is supported by a FRIA grant from the Fonds de la Recherche Scientifique de Belgique (F.R.S.-FNRS). Computational resources have been provided by the Consortium des Équipements de Calcul Intensif (CÉCI), funded by the Fonds de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under Grant $n°2.5020.11$ and by the Walloon Region. The present research also benefited from computational resources made available on the Tier-1 supercomputer of the Federation Wallonie-Bruxelles, infrastructure funded by the Walloon Region under the grant agreement $n°1117545$.

References

[1] R. Mittal, G. Iaccarino, Immersed boundary methods, Annual Review of Fluid Mechanics 37 (2005) 239–261.

[2] C. S. Peskin, Numerical analysis of blood flow in the heart, Journal of Computational Physics 25 (3) (1977) 220–252.

[3] K. Taira, T. Colonius, The immersed boundary method: A projection approach, Journal of Computational Physics 225 (2) (2007) 2118–2137.

[4] S. Liska, T. Colonius, A fast immersed boundary method for external incompressible viscous flows using lattice Green's functions, Journal of Computational Physics 331 (2017) 257 – 279.

[5] D. B. Stein, R. D. Guy, B. Thomases, Immersed boundary smooth extension: A high-order method for solving PDE on arbitrary smooth domains using Fourier spectral methods, Journal of Computational Physics 304 (2016) 252 – 274.

[6] P. Angot, C. Bruneau, P. Fabrie, A penalization method to take into account obstacles in incompressible viscous flows, Numerische Mathematik 81 (1999) 497–520.

[7] N. K.-R. Kevlahan, J.-M. Ghidaglia, Computation of turbulent flow past an array of cylinders using a spectral method with Brinkman penalization, European Journal of Mechanics - B/Fluids 20 (3) (2001) 333–350.

[8] F. Gibou, D. Hyde, R. Fedkiw, Sharp interface approaches and deep learning techniques for multiphase flows, Journal of Computational Physics 380 (2019) 442–463.

[9] R. J. Leveque, Z. L. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM Journal on Numerical Analysis 31 (4) (1994) 1019–1044.

[10] M. N. Linnick, H. F. Fasel, A high-order immersed interface method for simulating unsteady incompressible flows on irregular domains, Journal of Computational Physics 204 (2005) 157–192.

[11] Z. Li, I. Kazufumi, The immersed interface method: Numerical solutions of PDEs involving interfaces and irregular domains, SIAM, 2006.

[12] Y. Marichal, An immersed interface vortex particle-mesh method, Ph.D. thesis, Université catholique de Louvain, 2014.

[13] S. Hosseinverdi, H. F. Fasel, Very High-Order Accurate Sharp Immersed Interface Method: Application to Direct Numerical Simulations of Incompressible Flows, American Institute of Aeronautics and Astronautics, 2017.

[14] M. Gazzola, P. Chatelain, P. Koumoutsakos, Vortex methods for fluid-structure interaction problems with deforming geometries and their application to swimming, in: Division of Fluid Dynamics, American Physical Society, 2010.

[15] C. Mimeau, F. Gallizio, G.-H. Cottet, I. Mortazavi, Vortex penalization method for bluff body flows, International Journal for Numerical Methods in Fluids 79 (2) (2015) 55–83, fld.4038.

[16] M. M. Hejlesen, P. Koumoutsakos, A. Leonard, J. H. Walther, Iterative Brinkman penalization for remeshed vortex methods, Journal of Computational Physics 280 (2015) 547–562.

[17] T. Gillis, G. Winckelmans, P. Chatelain, An efficient iterative penalization method using recycled Krylov subspaces and its application to impulsively started flows, Journal of Computational Physics 347 (2017) 490 – 505.

[18] P. Koumoutsakos, A. Leonard, High-resolution simulations of the flow around an impulsively started cylinder using vortex methods, J. Fluid. Mech. 296 (1) (1995) 1–38.

[19] P. Ploumhans, G. S. Winckelmans, Vortex Methods for High-Resolution Simulations of Viscous Flow Past Bluff Bodies of General Geometry, Journal of Computational Physics 165 (2) (2000) 354–406.

[20] M. J. Lighthill, Introduction of boundary layer theory, in: R. L. (Ed.), Laminar Boundary Layers, Oxford University Press, New York, 1963.

[21] G.-H. Cottet, P. Poncet, Advances in direct numerical simulations of 3D wall-bounded flows by Vortex-in-Cell methods, J. Comput. Phys. 193 (1) (2004) 136–158.

[22] Y. Marichal, P. Chatelain, G. Winckelmans, An immersed interface solver for the 2-D unbounded Poisson equation and its application to potential flow, Computers & Fluids 96 (2014) 76–86.

[23] Y. Marichal, P. Chatelain, G. Winckelmans, Immersed interface interpolation schemes for particle–mesh methods, Journal of Computational Physics 326 (2016) 947 – 972.

[24] T. Gillis, G. Winckelmans, P. Chatelain, Fast immersed interface Poisson solver for 3D unbounded problems around arbitrary geometries, Journal of Computational Physics 354 (2018) 403 – 416.

[25] J. Sherman, W. J. Morrison, Adjustment of an Inverse Matrix Corresponding to a Change in One Element of a Given Matrix, Ann. Math. Statist. 21 (1) (1950) 124–127.

[26] W. W. Hager, Updating the Inverse of a Matrix, SIAM Review 31 (2) (1989) 221–239.

[27] R. Chatelin, P. Poncet, Hybrid grid-particle methods and Penalization: A Sherman-Morrison-Woodbury approach to compute 3D viscous flows using FFT, J. Comput. Phys. 269 (0) (2014) 314 – 328.

[28] G.-H. Cottet, P. Koumoutsakos, Vortex Methods, Theory and Practice, Cambridge University Press, 2000.

[29] G. S. Winckelmans, Encyclopedia of Computational Mechanics, chap. Vortex Methods, John Wiley & Sons, Ltd, ISBN 9780470091357, 129–153, 2004.

[30] J. Monaghan, Extrapolating B splines for interpolation, Journal of Computational Physics 60 (2) (1985) 253 – 262.

[31] P. Chatelain, A. Leonard, Isotropic compact interpolation schemes for particle methods, Journal of Computational Physics 227 (6) (2008) 3244 – 3259.

[32] P. Chatelain, A. Curioni, M. Bergdorf, D. Rossinelli, W. Andreoni, P. Koumoutsakos, Billion vortex particle Direct Numerical Simulations of aircraft wakes, Computer Methods in Applied Mechanics and Engineering 197 (13) (2008) 1296–1304.

[33] M. Hejlesen, J. Rasmussen, P. Chatelain, J. Walther, A high order solver for the unbounded Poisson equation, Journal of Computational Physics 252 (2013) 458–467.

[34] J. H. Williamson, Low-storage Runge-Kutta schemes, Journal of Computational Physics 35 (1) (1980) 48–56.

[35] P. Chatelain, P. Koumoutsakos, A Fourier-based elliptic solver for vortical flows with periodic and unbounded directions, Journal of Computational Physics 229 (7) (2010) 2425–2431.

[36] P.-G. Martinsson, G. J. Rodin, Asymptotic expansion of lattice Green's function, Proceedings: Mathematical, Physical and Engineering Sciences 458 (2027) (2002) 2609–2622.

[37] A. Gillman, P. Martinsson, A fast solver for Poisson problems on infinite regular lattices, Journal of Computational and Applied Mathematics 258 (2014) 42 – 56.

[38] A. Amritkar, E. Sturler, K. Swirydowicz, D. Tafti, K. Ahuja, Recycling Krylov subspaces for CFD applications and a new hybrid recycling solver, Journal of Computational Physics 303 (2015) 222–237.

[39] P. Ploumhans, G. Winckelmans, J. Salmon, A. Leonard, M. Warren, Vortex Methods for Direct Numerical Simulation of Three-Dimensional Bluff Body Flows: Application to the Sphere at Re=300, 500, and 1000, Journal of Computational Physics 178 (2) (2002) 427 – 463.

[40] T. D. Aslam, A partial differential equation approach to multidimensional extrapolation, Journal of Computational Physics 193 (1) (2004) 349 – 355.

[41] F. Noca, On the evaluation of instantaneous fluid-dynamic forces on a bluff body, Report FM96-5, GALCIT, 1996.

[42] M. Bar-Lev, H. T. Yang, Initial flow field over an impulsively started circular cylinder, Journal of Fluid Mechanics 72 (1975) 625–647.

[43] D. Rossinelli, B. Hejazialhosseini, W. van Rees, M. Gazzola, M. Bergdorf, P. Koumoutsakos, MRAG-I2D: Multi-resolution adapted grids for remeshed vortex methods on multicore architectures, Journal of Computational Physics 288 (2015) 1–18.

[44] R. A. James, Solution of Poisson's equation for isolated source distributions, Journal of Computational Physics 25 (2) (1977) 71–93.

[45] K. Lackner, Computation of ideal MHD equilibria, Computer Physics Communications 12 (1) (1976) 33 – 44.

[46] G. Miller, An iterative boundary potential method for the infinite domain Poisson problem with interior Dirichlet boundaries, Journal of Computational Physics 227 (16) (2008) 7917 – 7928.

[47] Y. Marichal, P. Chatelain, G. Winckelmans, An immersed interface vortex particle-mesh solver, in: Bulletin of the American Physical Society, vol. 59, American Physical Society, 2014.

[48] S. Liska, T. Colonius, A parallel fast multipole method for elliptic difference equations, Journal of Computational Physics 278 (2014) 76 – 91.

Appendix A. Polynomial interpolation

The Immersed Interface tool rely on interpolation coefficients to extrapolate/interpolate information from the flow up to the boundary. We here present the polynomial reconstruction of the information, as used in the present work.

Let us consider a general example with a set of 3 points (x_1, x_2 and x_3) with corresponding value (U_1, U_2 and U_3) and a boundary condition of degree k, $U_\alpha^{(k)}$, located at x_α. For our application, we look for the linear combination of $U_\alpha^{(k)}$, U_1, U_2 and U_3 to obtain the value of the dth derivative of the field, at x_α. We then build the following system using Taylor series and $\Delta_i = x_i - x_\alpha$,

$$c_1 U_1 + c_2 U_2 + c_3 U_3 + c_\alpha U_\alpha^{(k)} = u_\alpha \left[c_1 + c_2 + c_3 + \delta_{0k} c_\alpha\right] \tag{A.1}$$

$$+ u_\alpha^{(1)} \left[c_1 \left(\Delta_1\right) + c_2 \Delta_1 + c_3 \Delta_1 + \delta_{1k} c_\alpha\right] \tag{A.2}$$

$$+ u_\alpha^{(2)} \left[c_1 \frac{\left(\Delta_1\right)^2}{2} + c_2 \frac{\left(\Delta_2\right)^2}{2} + c_3 \frac{\left(\Delta_3\right)^2}{2} + \delta_{2k} c_\alpha\right] \tag{A.3}$$

$$+ u_\alpha^{(2)} \left[c_1 \frac{\left(\Delta_1\right)^3}{6} + c_2 \frac{\left(\Delta_2\right)^3}{6} + c_3 \frac{\left(\Delta_3\right)^3}{6} + \delta_{3k} c_\alpha\right] . \tag{A.4}$$

The accuracy of such a scheme is given by $\mathcal{O}\left(h^{4-q}\right)$. To obtain the interpolation coefficient, one solves the following system

$$\begin{bmatrix} \delta_{0k} & 1 & 1 & 1 \\ \delta_{1k} & \left(\Delta_1\right) & \left(\Delta_2\right) & \left(\Delta_3\right) \\ \delta_{2k} & \frac{\left(\Delta_1\right)^2}{2} & \frac{\left(\Delta_2\right)^2}{2} & \frac{\left(\Delta_3\right)^2}{2} \\ \delta_{3k} & \frac{\left(\Delta_1\right)^3}{6} & \frac{\left(\Delta_2\right)^3}{6} & \frac{\left(\Delta_3\right)^3}{6} \end{bmatrix} \begin{bmatrix} c_\alpha^q \\ c_1^q \\ c_2^q \\ c_3^q \end{bmatrix} = \begin{bmatrix} \delta_{0q} \\ \delta_{1q} \\ \delta_{2q} \\ \delta_{3q} \end{bmatrix} . \tag{A.5}$$

As an example, we detail the computation of the interpolation coefficients, S_i^q, with $k = 0$, as used in the IIVPM method. For each derivative, q, the coefficients are the solution of the following system:

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & \left(\Delta_1\right) & \left(\Delta_2\right) & \left(\Delta_3\right) \\ 0 & \frac{\left(\Delta_1\right)^2}{2} & \frac{\left(\Delta_2\right)^2}{2} & \frac{\left(\Delta_3\right)^2}{2} \\ 0 & \frac{\left(\Delta_1\right)^3}{6} & \frac{\left(\Delta_2\right)^3}{6} & \frac{\left(\Delta_3\right)^3}{6} \end{bmatrix} \begin{bmatrix} c_\alpha^q \\ c_1^q \\ c_2^q \\ c_3^q \end{bmatrix} = \begin{bmatrix} \delta_{0q} \\ \delta_{1q} \\ \delta_{2q} \\ \delta_{3q} \end{bmatrix} . \tag{A.6}$$

Appendix B. Lattice Green's function

In this section, we focus on the solution of a 2D unbounded Poisson problem,

$$\nabla^2 u = f \ , \tag{B.1}$$

where we approximate the Laplacian using the second order standard 3 point stencil, ∇_h^2. The straightforward inversion of ∇_h^2 operator is challenging and computationally intensive [44–47]. For the sake of efficiency, we rather follow the approach of a grid-based fast convolution of the Green's function with the source term, initially proposed Gillman and Martinsson [37] in 2D and Liska and Colonius [48] in 3D. In 2D, the Poisson equation expressed in wave space, $\left(k_x\ ;\ k_y\right) \in \left[-\pi/h_x\ ;\ \pi/h_x\right] \times \left[-\pi/h_y\ ;\ \pi/h_y\right]$, becomes

$$\frac{1}{h^2}\left[4\sin^2\left(k_x h_x/2\right) + 4\sin^2\left(k_y h_y/2\right)\right]\hat{u} \triangleq \sigma\left(\mathbf{k}h\right)\hat{u} = \hat{f} \quad , \tag{B.2}$$

where $\hat{u}$ and $\hat{f}$ are the Fourier transforms of u and f. At this step, we consider a structured grid with $h = h_x = h_y$. The associated lattice Green's function (LGF) is given by the unbounded inverse Fourier transform of $1/\left(\sigma\left(\mathbf{k}h\right)\right)$ [37]:

$$\Theta\left(\mathbf{x}_m\right) = \frac{1}{(2\pi)^2}\iint_{-\pi/h}^{\pi/h} \frac{h^2\ \ \exp\left(-\mathrm{i}\,\mathbf{x}_m\cdot\mathbf{k}\right)}{4\sin^2\left(k_x h/2\right) + 4\sin^2\left(k_y h/2\right)}\quad d\mathbf{k} \triangleq \Theta\left(\mathbf{m}\right) \quad . \tag{B.3}$$

Defining $\mathbf{m} = \dfrac{\mathbf{x}}{h} = \left\{m_x\ ;\ m_y\right\}$ and $\boldsymbol{\xi} = \mathbf{k}h$, we write

$$\Theta\left(\mathbf{m}\right) = \frac{1}{(2\pi)^2}\iint_{-\pi}^{\pi} \frac{\exp\left(-\mathrm{i}\,\mathbf{m}\cdot\boldsymbol{\xi}\right)}{4\sin^2\left(\xi_x/2\right) + 4\sin^2\left(\xi_y/2\right)}\quad d\boldsymbol{\xi} \quad . \tag{B.4}$$

For small $\left|\mathbf{m}\right|$, we evaluate this integral; for $\left|\mathbf{m}\right|$ sufficiently large (in practice, $\left|\mathbf{m}\right| \geq 8$), Eq. (B.4) is replaced by the following expansion [36]

$$\begin{aligned}
\Theta\left(\mathbf{m}\right) = {} & \frac{1}{2\pi}\left[\log\left(\mathbf{m}\right) + \gamma + \frac{\log\left(8\right)}{2}\right] \\
& - \frac{1}{24\pi\left|\mathbf{m}\right|^6}\left[m_1^4 + m_2^4 - 6m_1^2m_2^2\right] \\
& - \frac{1}{480\pi\left|\mathbf{m}\right|^{12}}\left[43\ m_1^8 - 772\ m_1^6m_2^2 + 1570\ m_1^4m_2^4 - 772\ m_1^2m_2^6 + 43\ m_2^8\right] \\
& - \frac{1}{2016\pi\left|\mathbf{m}\right|^{18}}\left[609\ m_1^{12} - 24234\ m_1^{10}m_2^2 + 109935\ m_1^8m_2^4\right. \\
& \qquad \left. - 160524\ m_1^6m_2^6 + 109935\ m_1^4m_2^8 - 24234\ m_1^2m_2^{10} + 609\ m_2^{12}\right] \\
& - \frac{1}{2880\pi\left|\mathbf{m}\right|^{24}}\left[63139\ m_1^{16} - 4467336\ m_1^{14}m_2^2 + 38334996\ m_1^{12}m_2^4 - 98512568\ m_1^{10}m_2^6\right. \\
& \qquad \left. + 122747922\ m_1^8m_2^8 - 98512568\ m_1^6m_2^{10} + 38334996\ m_1^4m_2^{12} - 4467336\ m_1^2m_2^{14} + 63139\ m_2^{16}\right] \\
& + \mathcal{O}\left(1/\left|\mathbf{m}\right|^{10}\right)
\end{aligned} \tag{B.5}$$

Using this LGF, we express the application of the inverse of ∇_h^2 as

$$u = \left[\nabla_h^{-2} f\right]_{ij} = \sum_{kl} \Theta\left(\mathbf{x}_{kl} - \mathbf{x}_{ij}\right) f\left(\mathbf{x}_{kl}\right) \quad , \tag{B.6}$$

a convolution that can be computed using a FFT-based [24] or kiFMM algorithm [37].

Appendix C. Summary of the Immersed Interface stencil used

In this section, we list the two encountered cases and the interpolation stencils that we use for every sub-step of the Immersed Interface method.

- **First derivative - Dirichlet boundary condition**: wall vorticity

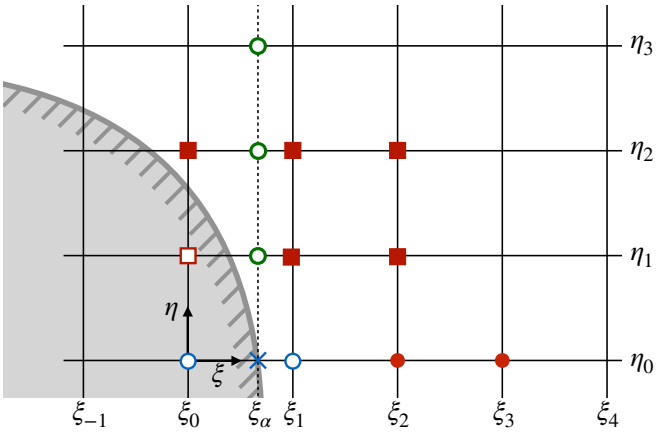


Figure C.19: **IIM stencils** - wall vorticity third order stencils

$$\left.\frac{\partial^k u}{\partial \xi^k}\right|_\alpha = S_\alpha^{\xi,k}\, u_\alpha + \sum_{i=1}^{2} S_i^{\xi,k}\, u\left(\xi_{i+1}, \eta_0\right) + \mathcal{O}\left(h^{3-k}\right) \tag{C.1}$$

$$\left.\frac{\partial^k u}{\partial \eta^k}\right|_\alpha = S_\alpha^{\eta,k}\, u_\alpha + \sum_{i=1}^{2} S_i^{\eta,k}\, u\left(\xi_\alpha, \eta_i\right) + \mathcal{O}\left(h^{3-k}\right) \tag{C.2}$$

$$u\left(\xi_\alpha, \eta_j\right) = \sum_{i=1}^{3} T_i^{\xi,0}\, u\left(\xi_{i-2+s_{\eta_j}}, \eta_j\right) + \mathcal{O}\left(h^3\right) \tag{C.3}$$

- **Second derivative - Dirichlet boundary condition**: Poisson , diffusion

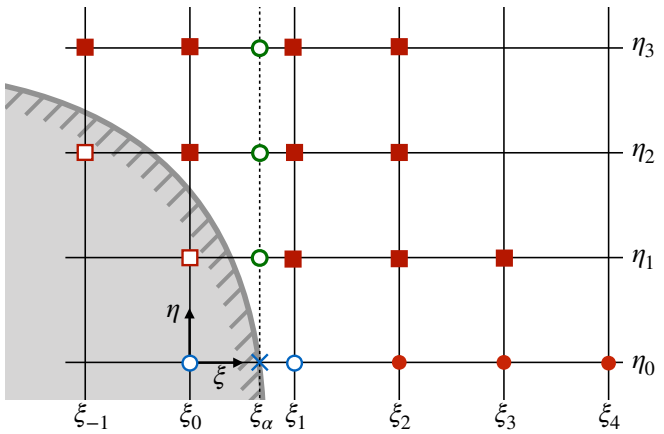


Figure C.20: **IIM stencils** - Poisson and diffusion

$$\left.\frac{\partial^k u}{\partial \xi^k}\right|_\alpha = S_\alpha^{\xi,k}\, u_\alpha + \sum_{i=1}^{3} S_i^{\xi,k}\, u\left(\xi_{i+1}, \eta_0\right) + \mathcal{O}\left(h^{4-k}\right) \tag{C.4}$$